พิมพ์ภาพกราฟิกด้วย ปาสคาล 4.0

อาจหาญ สัตยารักษ์

การเขียนภาพกราฟิกนั้นเป็นเรื่องง่ายโดยเฉพาะในปาสคาล 4.0 ที่มีฟังก์ชันด้านกราฟิกให้อย่างมากมายอยู่แล้ว จึงคิดว่าผู้อ่านก็คงจะใช้เขียนกราฟิกเป็นประจำอยู่แล้ว แต่การพิมพ์ภาพที่ได้ลงกระดาษนั้นต่างหากที่เป็นเรื่องยาก หรืออาจจะเป็นเรื่องง่ายในกรณีที่ท่านมีโปรแกรมใช้งานอยู่ แต่ท่านก็คงอยากทราบว่าการพิมพ์ภาพมีหลักการของการทำงานในโปรแกรมนั้น ๆ ว่าอย่างไร หรืออาจใช้งานได้เฉพาะเมื่อท่านใช้กราฟิกบนจอสีแต่ไม่สามารถพิมพ์ภาพจากจอโมโนโครมได้ เป็นต้น ทั้งนี้เพราะเป็นโปรแกรมที่ตรวจสอบหน่วยความจำของภาพมาพิมพ์ เมื่อภาพบนจอสีและจอโมโนโครมอยู่คนละตำแหน่งกันโปรแกรมนั้น ๆ ก็ไม่สามารถทำงานได้

โปรแกรมที่ผู้เขียนจะเสนอในครั้งนี้ไม่สนใจตำแหน่งในหน่วยความจำ แต่สนใจตำแหน่งของภาพเป็นสำคัญ ทั้งนี้เพราะผู้เขียนใช้วิธีตรวจที่ตำแหน่งของภาพเลยว่าลงจุดหรือไม่ (ได้ค่าเป็น 0 ถ้าไม่ลงจุด และมีค่าไม่เท่ากับ 0 เมื่อลงจุด) คำสั่งนั้นคือ **GetPixel(x,y : Integer);** *ซึ่งจะตรวจว่าตรงจุด x, y นั้นมีการลงจุดหรือไม่*

ใช้ความรู้ด้านเครื่องพิมพ์มาช่วยงาน

คำสั่งของเครื่องพิมพ์ต่อไปนี้ ผู้เขียนจะใช้ของเครื่องพิมพ์ EPSON ซึ่งคำสั่งที่ใช้เกือบทุกรุ่นจะเหมือนกัน (ท่านที่มีเครื่องพิมพ์อื่นอาจทดลองใช้ดูก่อน ถ้าไม่ทำงานค่อยอ่านคู่มือของเครื่องพิมพ์ก็คงแก้ไขให้ทำงานได้ เท่าที่ตรวจดูจากคู่มือคิดว่าเครื่อง NEC ก็คงจะใช้ได้) คำสั่งที่ใช้จะมีดังนี้

ESC 3 ใช้จัดระยะบรรทัดของเครื่องพิมพ์ คำสั่งภาษาปาสคาลแบบง่าย ๆ ก็คือ Writeln (Lst, chr(27)+'3'+chr(n); เมื่อ n คือตัวเลขที่เราต้องการให้ได้ระยะบรรทัด (n/216 นิ้ว แต่ของ NEC เป็น n/180 นิ้ว)

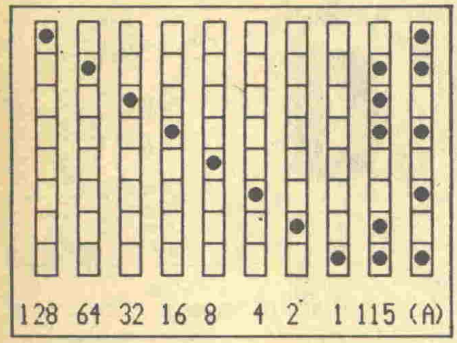
ESC* สำหรับกำหนดรูปแบบการพิมพ์ให้แก่เครื่องพิมพ์โดยกำหนดเป็นรหัส 8 บิต (0–255) คำสั่งคือ Writeln (Lst, #27+'*'+chr(m)+chr(n1)+chr(n2); โดยค่า m มีค่าจาก 0 ถึง 7 ส่วนค่า n1, n2 กรุณาศึกษาจากคู่มือเครื่องพิมพ์ (ซึ่งส่วนมากเรามักจะไม่สนใจอ่านกันมากนัก ทั้งที่เป็นสิ่งมีประโยชน์ต่อการใช้เครื่องพิมพ์)

ESC 1 สำหรับกำหนดกั้นซ้าย (left margin) คือ Writeln(Lst, #27+'1'+chr(n); ค่า n จะขึ้นกับชนิดรุ่นของเครื่องพิมพ์ เราจะใช้กำหนดให้ภาพเริ่มพิมพ์ในตำแหน่งที่เหมาะสม

แต่สำหรับการทำงานในที่นี้ผู้เขียนใช้วิธีกำหนดค่ารีจิสเตอร์ส่งผ่านไปให้เครื่องพิมพ์ทำงานอีกทีเพราะจะสะดวกกว่า (อยู่ในชุดคำสั่งชื่อ **CopyOneByte** ในรูปที่ 3)

เรามาดูค่ารีจิสเตอร์ที่ส่งไปให้เครื่องพิมพ์ว่าสัมพันธ์กับหัวพิมพ์อย่างไรได้จากรูปที่ 1

ดังนั้นถ้าต้องการให้เข็มของเครื่องพิมพ์ทำงานตามรูป A ก็ต้องใช้ค่า 128+64+16+4+1 ซึ่งก็คือค่า 213 ในทำนองเดียวกันถ้าเราต้องการให้เครื่องพิมพ์พิมพ์ตรงข้ามกับ A ก็ให้ค่า 32+8+2 = 42 ซึ่งก็คือ 255–213 = 42 นั่นเอง โดยเราจะใช้ประโยชน์นี้ไปสร้างภาพได้ 2 แบบคือ แบบปกติ



รูปที่ 1 แสดงค่ารีจิสเตอร์ที่ใช้ควบคุมหัวเข็มแต่ละเล่ม

และแบบกลับขาวดำ

ลองมาดูส่วนของการสร้างกราฟิกกันบ้าง

ในที่นี้ขอใช้กรอบกราฟิกของจอโมโนโครมคือขนาด 720×350 จุด มาอธิบายว่าเราจะนำไปพิมพ์ได้อย่างไร เราก็ใช้วิธีง่าย ๆ คือ

1. แบ่งส่วนของจอภาพออกเป็นส่วน ๆ ละ 8 แถวเพื่อใช้ตรวจหาการลงจุด ดังนั้นในจอโมโนโครมจะแบ่งได้ 44 ส่วน

2. แต่ละส่วนแบ่งเป็นแนวตั้งจากตำแหน่งซ้ายสุดไปขวาสุด แนวละ 8 แถว ซึ่งจะได้ 720 แนวตั้ง

3. นำแนวตั้งแต่ละแนวนั้นมาตรวจหาการลงจุด เพื่อคำนวณหาค่ารีจิสเตอร์ส่งไปให้เครื่องพิมพ์ เช่น ถ้าได้แบบ A ในรูปที่ 1 ก็จะได้ 213 เป็นต้น

4. ส่งค่าที่หาได้นี้ไปพิมพ์ทีละแถว ๆ ละ 720 แนวตั้ง เมื่อพิมพ์ครบ 44 แถวก็จะได้ภาพทั้งหมดตามต้องการ

นี่คือการพิมพ์ตามปกติซึ่งจะแบ่งส่วนของจอภาพได้ดังรูปที่ 2

ซึ่งตามรูปเป็นการพิมพ์ภาพให้เหมือนที่เห็นบนจอภาพเลย ถ้าต้องการพิมพ์อีกด้านหนึ่งก็กำหนดใหม่ ซึ่งผู้เขียนได้เขียนโปรแกรมไว้แล้ว ศึกษาได้จากโปรแกรมในรูปที่ 3

เพื่อสะดวกในการใช้โปรแกรมพิมพ์ภาพกราฟิกดังกล่าว จึงได้เขียนเป็นชุดคำสั่งย่อยซ้อนอยู่ในชุดคำสั่ง **WaitToGo** กล่าวคือเมื่อแสดงกราฟิกหมดหนึ่งกรอบก็ให้เรียกชุดคำสั่งนี้ทำงาน ซึ่งจะเห็นว่ามีทางเลือกได้หลายทาง (อยู่ที่ส่วนท้ายสุดของโปรแกรม) ดังนี้

1. กดแป้น ESC จะออกจากกราฟิกเข้าสู่เมนูหลักของปาสคาล 4.0

2. กดแป้น I จะเป็นการกำหนดการพิมพ์แบบกลับขาวดำ

3. กดแป้น Shift – F1 ถึง Shift – F3 จะพิมพ์ภาพได้ 3 แบบ รวมทั้งแบบกลับขาวดำ ก็จะได้ทั้งหมด 6 แบบ

รูปที่ 2 แสดงการแบ่งส่วนของกราฟิกออกเป็นส่วน ๆ

แถวที่ 0 Y = 0 ถึง 7	Y จึงเริ่มที่ 0*8 จบที่ 0*8+7
แถวที่ 1 Y = 8 ถึง 15	Y จึงเริ่มที่ 1*8 จบที่ 1*8+7
แถวที่ 2 Y = 16 ถึง 23	Y จึงเริ่มที่ 2*8 จบที่ 2*8+7
แถวอื่น ๆ อีกตามที่คำนวณได้	
XMin = 0	XMax = 719

รูปที่ 3 โปรแกรมพิมพ์ภาพกราฟิกจากจอลงเครื่องพิมพ์

```
PROCEDURE  WaitToGo;
  var
    Ch :char;
    Inverse :word;
    CodeforExit :byte;

  PROCEDURE CopyGraphics(PrtCode:byte);
   const
    LeftMarginS :array[1..3] of byte=(10,0,33);
    SMode :array[1..3] of byte=(1,1,4);
   var
    ScanLine : integer;
    n1,n2,LeftMargin,Mode : byte;
    SCanmax,EndScan :word;

    PROCEDURE CopyOneByte(OneByte : byte);
       var
        Regs : Registers;
      begin
        Regs.AH := 0;
        Regs.AL := OneByte;
        Regs.DX := Pred(1);
        Intr($17, Regs);
      end; { CopyOneByte }

    PROCEDURE Copy8Line;
      const
      Bits : array[0..7] of byte = (128,64,32,16,8,4,2,1);
       var
        Bit, PrintByte: byte;
        XYPixel,XY: integer;
      begin
        CopyOneByte(27);
        CopyOneByte(Ord('*'));
        CopyOneByte(Mode);
        CopyOneByte(n1);
        CopyOneByte(n2);
      for XYPixel := 0 to ScanMax do
       begin
         XY := Scanline * 8;
         PrintByte := 0;
      for Bit := 0 to 7 do
       begin
        if (PrtCode=1) and (GetPixel(XYPixel,XY+bit) <> 0) then
              PrintByte:= PrintByte + Bits[Bit];
```

(ผู้เขียนเลือกวิธีกด 2 แป้นพิมพ์เพื่อกันความพลั้งเผลอที่อาจเกิดขึ้น เพราะการกด 2 แป้นพร้อมกันโอกาสที่จะพลาดนั้นค่อนข้างยาก หากไม่พอใจจะเปลี่ยนค่า #84 ถึง #86 ไปเป็นค่าอื่นก็ได้ โดยหาได้จากคู่มือของปาสคาลเอง)

4. กดแป้นอักขระอื่น ๆ ก็จะกลับมาทำงานต่อไป (เมื่อ CodeForExit = 27)

โปรแกรมส่วนอื่น ๆ

ชุดคำสั่ง **CopyGraphics** คือส่วนทั้งหมดของโปรแกรมการพิมพ์ภาพ โดยมีชุดคำสั่งย่อยซ้อนอยู่ข้างในดังนี้

1. ชุดคำสั่ง **CopyOneByte** คือส่วนที่รับค่ารีจิสเตอร์ที่คำนวณได้หรือกำหนดส่งไปยังเครื่องพิมพ์

2. ชุดคำสั่ง **Copy8Line** คือชุดคำสั่งที่ตรวจสอบตำแหน่งต่าง ๆ ของจอภาพเพื่อส่งไปให้เครื่องพิมพ์ทำงานให้ชื่อตามที่โปรแกรมจะพิมพ์ภาพทีละ 8 แถวของกราฟิกนั่นเอง โดยต้องส่งมาตรฐานของเครื่องพิมพ์ไปให้รับรู้ก่อน นั่นคือคำสั่ง ESC* นั่นเอง

การพิมพ์จะตรวจทีละ 8 แถว (ค่า ScanLine) และแต่ละจุดตามแกนตั้ง (XyPixel) โดยค่าในแกนตั้งจะเพิ่มทีละ 1 แถว (ตัวอย่างเช่น ส่วนแรกจากแถวที่ 0 ไปแถวที่ 7 เป็นต้น) เมื่อมีการลงจุด (ค่า GetPixel <> 0) ก็จะได้ค่า PrintByte:=PrintByte + Bits[bit]; ซึ่งได้กำหนดค่าของ Bits[0...7] ไว้ตามแนวคิดจากรูปที่ 1

ถ้าต้องการได้ภาพกลับขาวดำ (Inverse = 1) ก็ให้นำค่า PrintByte ไปลบออกจาก 255 ตามโปรแกรมที่เขียนว่า PrintByte:=255 – PrintByte;

แล้วส่งค่าที่ได้นี้ไปยังชุดคำสั่ง CopyOneByte เมื่อจบหนึ่งบรรทัด หรือหนึ่งส่วนของภาพ ก็ส่งรหัส 10 (CopyOneByte(10)); เพื่อให้จบบรรทัด

3. ชุดคำสั่ง CopyGraphics จริง ๆ นั้นเริ่มที่การหารูปแบบในการพิมพ์ (คือค่า Mode, Leftmargin) กำหนดค่า ScanMax, EndScan โดยถ้าพิมพ์ตามยาวของกระดาษ (PrtMode <>1) จะได้ค่ากลับกัน

จากนั้นส่งรหัส #27+'3'+#24 ไปจัดบรรทัดของเครื่องพิมพ์เพื่อจะได้ภาพกราฟิกที่ต่อเนื่องกันพอดี คำนวณหาค่า n1, n2 ที่เหมาะสม แล้วเริ่มพิมพ์จากส่วนที่ 0 (ScanLine = 0) ไปจนถึงส่วนสุดท้ายที่คำนวณได้จากค่า EndScan div 8 โดยใช้งานพร้อมกับชุดคำสั่ง Copy8Line ที่ได้อธิบายไปแล้ว และจบการพิมพ์กราฟิกด้วยการส่งรหัส ESC 2 แก่เครื่องพิมพ์เพื่อให้ระยะบรรทัดมาเป็น 1/6 นิ้วตามเดิม และส่งรหัส ESC + '1' + '0' เพื่อจัดกั้นซ้ายให้กลับมายังซ้ายสุด กับส่งรหัส ESC< เพื่อเลื่อนหัวเครื่องพิมพ์ไปที่คอลัมน์แรกอันเป็นจุดสิ้นสุดขั้นตอนการพิมพ์ภาพกราฟิก

ลองมาพิจารณากราฟิกกันสักหน่อย

ที่นี้ต้องลองดูว่าโปรแกรมนี้ทำงานได้ตามต้องการหรือไม่ โดยการเขียนโปรแกรมกราฟิกมารัน แล้วเรียกชุด

```
    if (PrtCode<>1) and (GetPixel(XY+bit,ScanMax-XYPixel) <> 0) then
          PrintByte := PrintByte+ Bits[Bit];
    if Inverse=1 then PrintByte:=255-PrintByte;
   end;
     CopyOneByte(PrintByte);
   end;
 CopyOneByte(10);
end; { Copy8Line }
 BEGIN  { CopyGraphics }
    Mode:=SMode[PrtCode];
    LeftMargin:=LeftMarginS[PrtCode];
    ScanMax:=719;  EndScan:=349;
   if PrtCode>1 then begin ScanMax:=349; EndScan:=719; end;
     CopyOneByte(27);
     CopyOneByte(Ord('3'));
     CopyOneByte(24);
   n1 := Lo(Succ(ScanMax));
   n2 := Hi(Succ(ScanMax));
     CopyOneByte(27);
     CopyOneByte(ord('1'));
     CopyOneByte(LeftMargin);
   for ScanLine := 0 to (EndScan div 8) do
          Copy8Line;
     CopyOneByte(27); CopyOneByte(2);
     CopyOneByte(27);
     CopyOneByte(ord('1'));
     CopyOneByte(0);
     CopyOneByte(27); CopyOneByte(ord('<'));
 END; { CopyGraphics }

BEGIN { WaitToGo }
 CodeforExit:=0; Inverse:=0;
 repeat
  Ch := ReadKey;
  if Ch = #27 then begin CloseGraph; Halt(0);  end;
  if UpCase(Ch)='I' then begin Inverse:=1; Ch:=#7;  end;
  if Ch=#0 then begin
     ch:= readKey;
    if Ch=#84 then CopyGraphics(1);
    if Ch=#85 then CopyGraphics(2);
    if Ch=#86 then CopyGraphics(3);
              end;  { Ch=#0 }
  if (Ch>#31) and (Ch<#127) then CodeforExit:=27;
 until CodeforExit=27;
END; { WaitToGo }
```

```
PROCEDURE TextShow;
BEGIN
  rectangle(130,0,580,86);
 OutTextThai(135,47,'ภาษาไทยกราฟิก',3,3);
 OutTextThai(144,79,'/โดย อาจหาญ สัตยารักษ์',2,1);
 OutTextThai(40,132,'\!ยินดีต้อนรับทุกท่าน!',2,1);
 OutTextTE(80,170,'ที่มาใช้ภาษาไทยและภาษาอังกฤษกราฟิก บน ปาสคาล 4.0',1);
 OutTextTE(80,200,'ซึ่งผู้เขียนได้พัฒนาขึ้นมานี้',1);
 OutTextThai(80,230,'_มียูนิตสร้างใหม่ 2 ยูนิต คือ Thai และ ThaiBig_',1,1);
 OutTextThai(9,305,'/! มีปัญหาติดต่อที่ !',2,4);
 QuadPoly(340,290,334,305,390,305,390,290);
 moveto(390,315); lineto(390,280); lineto(412,298); lineto(390,315);
 floodfill(400,298,blue); rectangle(412,240,717,330);
 rectangle(413,241,716,329); rectangle(414,242,715,328);
 OutTextTE(458,265,'โรงเรียนลำปางกัลยาณี',2);
 OutTextTE(458,290,'อำเภอเมือง ลำปาง',2);
 OutTextXY(458,300,'52000');
 End ; {TextShow }

{================== CopyGraphics ====================}

begin
  OpenGraph;
    TextShow;
    WaitToGo;
  CloseGraph;
end.
```

รูปที่ 4 โปรแกรมวาดภาพกราฟิกตัวอย่างเพื่อจะพิมพ์ลงกระดาษ

รูปที่ 5 ภาพที่ได้จากโปรแกรมในรูปที่ 4 ด้วยการกดแป้น Shift – F1

คำสั่ง WaitToGo ก็จะได้ภาพลงเครื่องพิมพ์ตามที่เราต้องการ

โปรแกรมในรูปที่ 4 นั้นคือตัวอย่างภาพกราฟิกที่จะพิมพ์ให้ดูเป็นตัวอย่าง ผู้อ่านอาจจะแปลกใจที่เห็นคำสั่งในชุดคำสั่ง TextShow เพราะมีบางคำสั่งไม่ใช่คำสั่งของปาสคาล 4.0 เดิม แต่เป็นคำสั่งที่ผู้เขียนสร้างขึ้นมาใช้ในกราฟิกภาษาไทยเอง (ถ้าสนใจก็ติดต่อได้ที่ผู้เขียน) ดังนั้นคงต้องลองเขียนคำสั่งกราฟิกอื่น ๆ มารันดูเอง คำสั่งที่แปลก ๆ นั้นมี OutTextTE, OutTextThai เป็นต้น โปรแกรมส่วนนี้ใช้งานคู่กับโปรแกรมในรูปที่ 3 คือชุดคำสั่ง WaitToGo พร้อมกับเขียนที่บรรทัดแรกเพิ่ม คือ

USES crt, printer, graph, Thai, ThaiBig; (ยูนิต Thai, ThaiBig เป็นยูนิตที่ใช้แสดงผลภาษาไทยที่ผู้เขียนพัฒนาขึ้นมาใช้ในงานกราฟิก เขียนตัวใหญ่ได้หลายขนาด) ซึ่งเป็นการใช้งานด้านกราฟิกตามปกติของปาสคาล 4.0

เมื่อปรากฏภาพขึ้นบนจอและกำลังอยู่ในส่วนของชุดคำสั่ง WaitToGo ก็กดแป้น Shift – F1 จะได้ภาพออกมาเหมือนรูปที่ 5 ทุกประการ ลองดูภาพของท่านเองก็จะรู้ว่าภาพที่พิมพ์จากการกดแป้นอื่น ๆ (Shift – F2, Shift – F3, I) จะให้ผลบนเครื่องพิมพ์เป็นอย่างไร บอกเพียงคร่าว ๆ ว่าภาพที่ได้จากการกดแป้นที่เหลือนั้นจะสามารถพิมพ์รวมกันบนกระดาษหนึ่งแผ่นได้พอดี โดยมีช่วงที่กด Shift – F2 อยู่ทางด้านซ้ายและที่เหลืออยู่ทางด้านขวา หรือจะลองพิมพ์ภาพกลับขาวดำดูก็ได้

หวังว่าท่านคงจะไม่มีปัญหาในการพิมพ์ภาพกราฟิกบนปาสคาล 4.0 อีกต่อไป ผู้อ่านที่ใช้กราฟิกบนตัวแปรภาษาอื่น ๆ ก็คงจะใช้หลักการนี้ได้เช่นเดียวกัน ขอให้ทุกท่านสนุกกับการใช้เครื่องพิมพ์ให้มากยิ่งขึ้น